Advanced Regression: 4a Machine learning: Classification

Garyfallos Konstantinoudis

Epidemiology and Biostatistics, Imperial College London

14th March 2023

Classification of binary outcomes

- Assume we have a binary outcome y_i for subject i.

$$y_i = \begin{cases} 1 \text{ if subject } i \text{ is a case} \\ 0 \text{ if subject } i \text{ is a control} \end{cases}$$

- Additionally, we have for each subject a predictor matrix x_i including p predictors or features.

$K = 2$ Classification of two classes

Aim is to predict the outcome of a new observation i based on x_i and to assign observation i to either class $k = 0$ or $k = 1$.
Decisions are based on the probability to belong to class k conditional on x_i.

$$\mathbb{P}(y_i = k \mid x_i)$$

Evaluation of classification performance: Confusion matrix

		Unobserved truth	
		Negative	Positive
Declared	Negatives	True Negative (TN)	False negative (FN)
	Positives	False positive (FP)	True Positive (TP)
Total		N	P

Confusion matrix:

Collection of TP, FP, TN, FN of a classifier.

Characteristics derived from the confusion matrix

- **Sensitivity**, recall, or true positive rate (TPR):

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN}$$

- **Specificity**, or true negative rate (TNR):

$$TNR = \frac{TN}{N} = \frac{TN}{TN + FP} = 1 - FPR$$

- **False positive rate** (FPR):

$$FPR = \frac{FP}{N} = \frac{FP}{TN + FP} = 1 - TNR$$

- **Precision**, or positive predictive value (PPV):

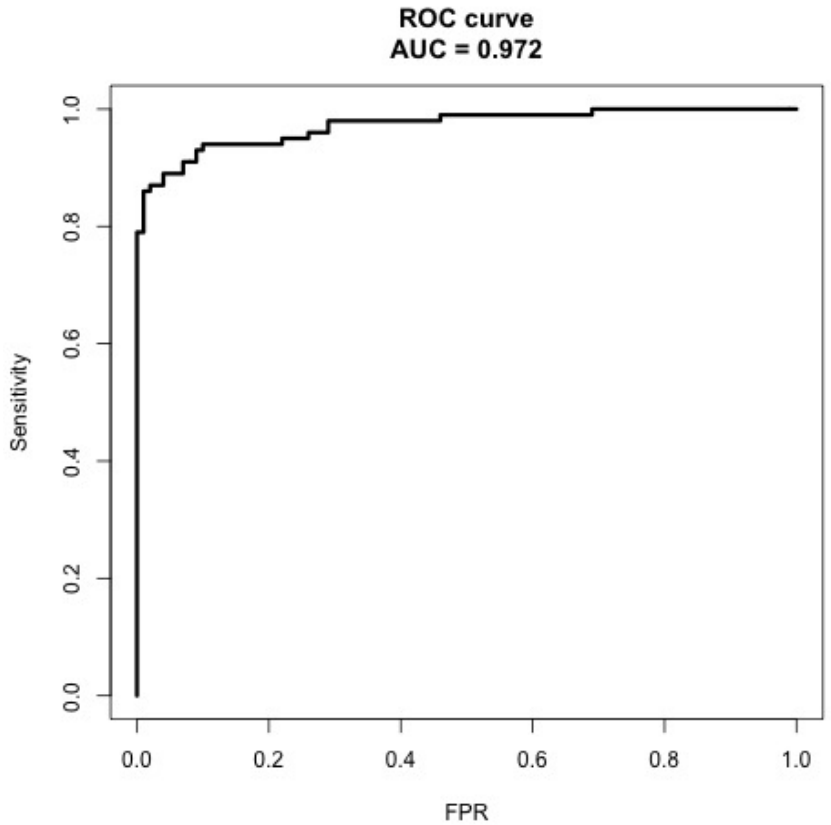
$$PPV = \frac{TP}{\text{Declared } P} = \frac{TP}{TP + FP} = 1 - FDR$$

Interpretation

- **Sensitivity**, or true positive rate (TPR): The probability that the test result picks up the disease.
- **Specificity**, or true negative rate (TNR):The probability that the test result identifies the ones that do not have the disease.
- **False positive rate** (FPR): the expectancy of the false positive ratio, i.e. how likely is to have a negative test given that the patient has the disease.
- **Precision**, or positive predictive value (PPV): how likely it is for someone to truly have the disease, in case of a positive test result.

Receiver operating characteristic (ROC)

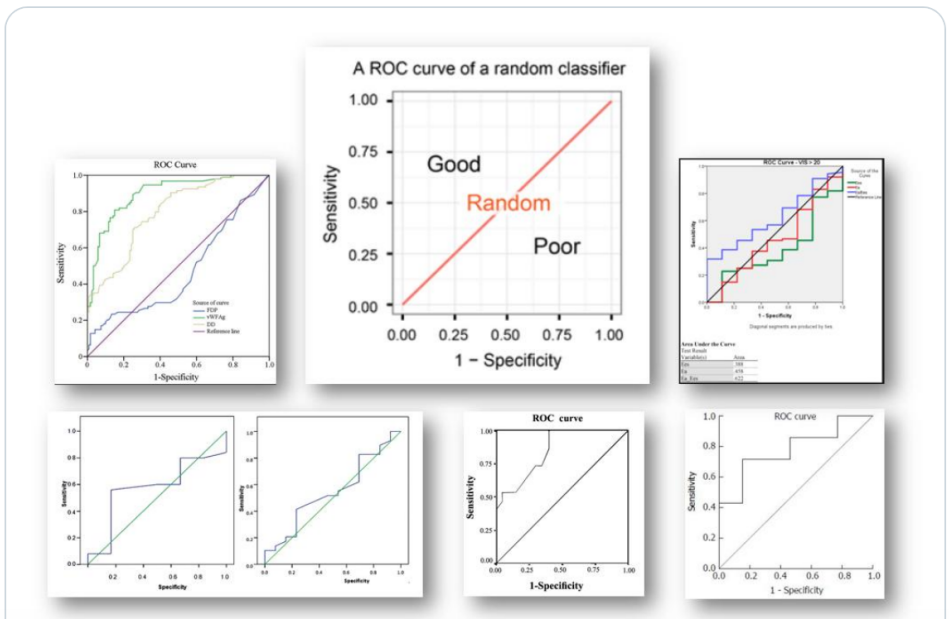
- Plot sensitivity (TPR) on the y-axis against 1-specificity (FPR) on the x-axis.
- Area under the ROC curve as summary-statistic for prediction performance.



Cecile Janssens
@cecilejanssens

The area under the ROC curve (AUC) is so frequently criticized and misunderstood that I often wonder whether I am the metric's only fan. Let me explain why how I see and value the AUC.

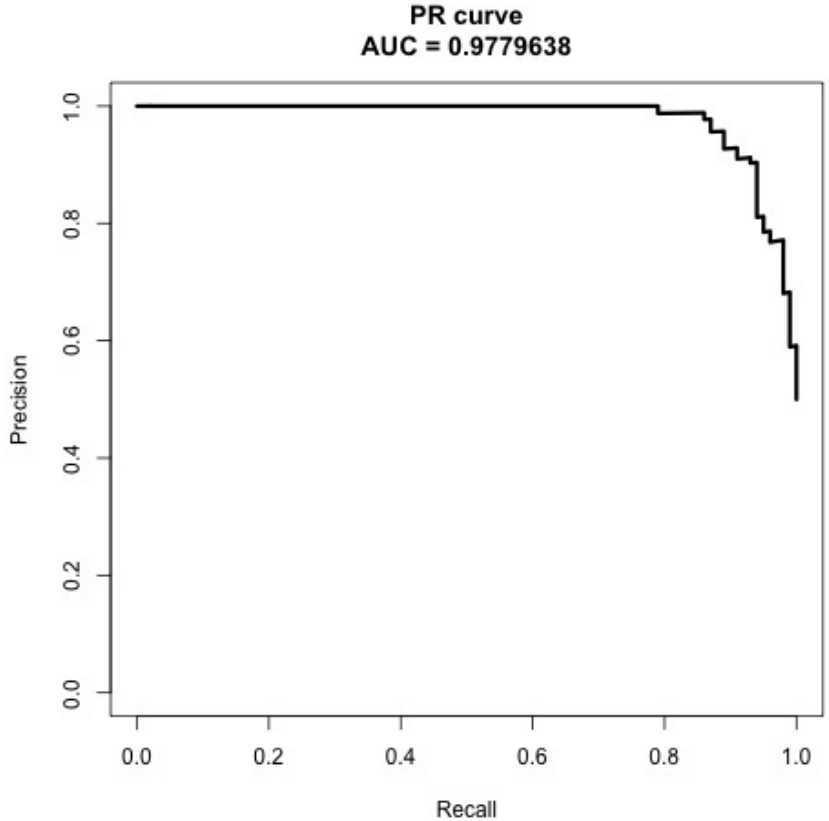
(thread)



- Tweetorial on AUC `https://twitter.com/cecilejanssens/status/1104134423673479169`

Precision-recall curve

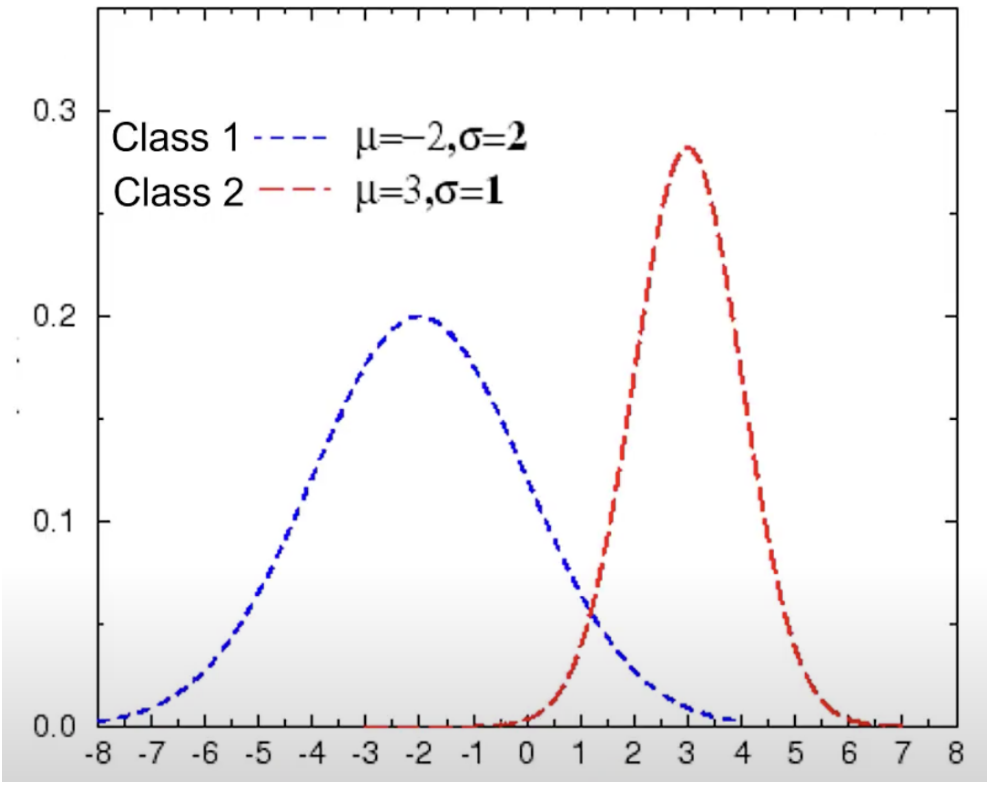
- Plot precision or positive predictive value (PPV) on the y-axis against the recall or true positive rate (TPR) on the x-axis.



ROC and precision-recall curve in R

1. Package `pROC`:
   - ROC plot using the function `roc(true.y,predicted.y)`
2. Package `PRROC`:
   - ROC plot using the function `roc.curve(fg,bg,curve=TRUE)`
   - Precision-recall plot using the function `pr.curve(fg,bg,curve=TRUE)`
   - Note: fg is a prediction score for group 1 and bg is a prediction score for group 0.

Motivation of discriminant analysis

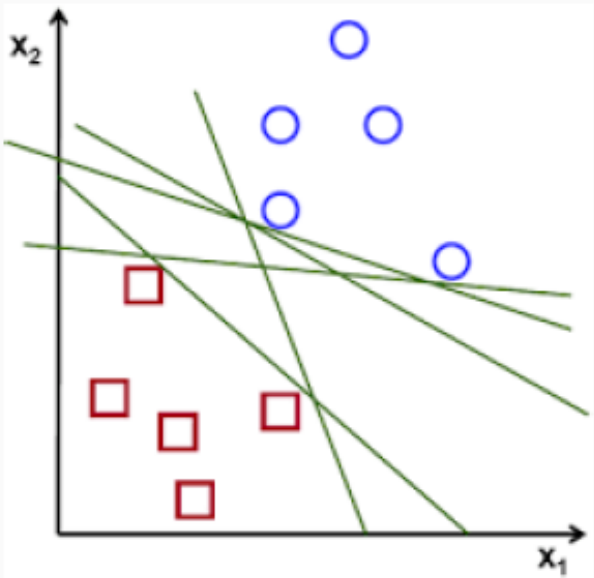


Example of classification rule: Assign the new value x to the class with the greatest likelihood

$$\hat{y} = \text{argmax} L(\mu_i, \sigma_i | x)$$

Motivation of discriminant analysis

- DA partitions $X = R^d$ into regions with the same class predictions via separating hyperplanes.
- Fit a Gaussian distribution for each class
- Find the line where the probability of being in both classes is the same.



Motivation of discriminant analysis

Derive the posterior probability for subject i to belong to class k based on:

- **Likelihood** of the data for $p = 1$ predictor (univariate Gaussian distribution) conditional on class k

$$f(x \mid k) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left\{-\frac{1}{2}\left(\frac{x - \mu_k}{\sigma_k}\right)^2\right\}$$

  where
  - μ_k is the expectation of x in group k and
  - σ_k^2 the variance of x in group k.
  - Note that each class $k = 0$ or $k = 1$ has its specific mean μ_k and variance σ_k^2.
- **Prior probability** of belonging to class k is π_k

Motivation of discriminant analysis

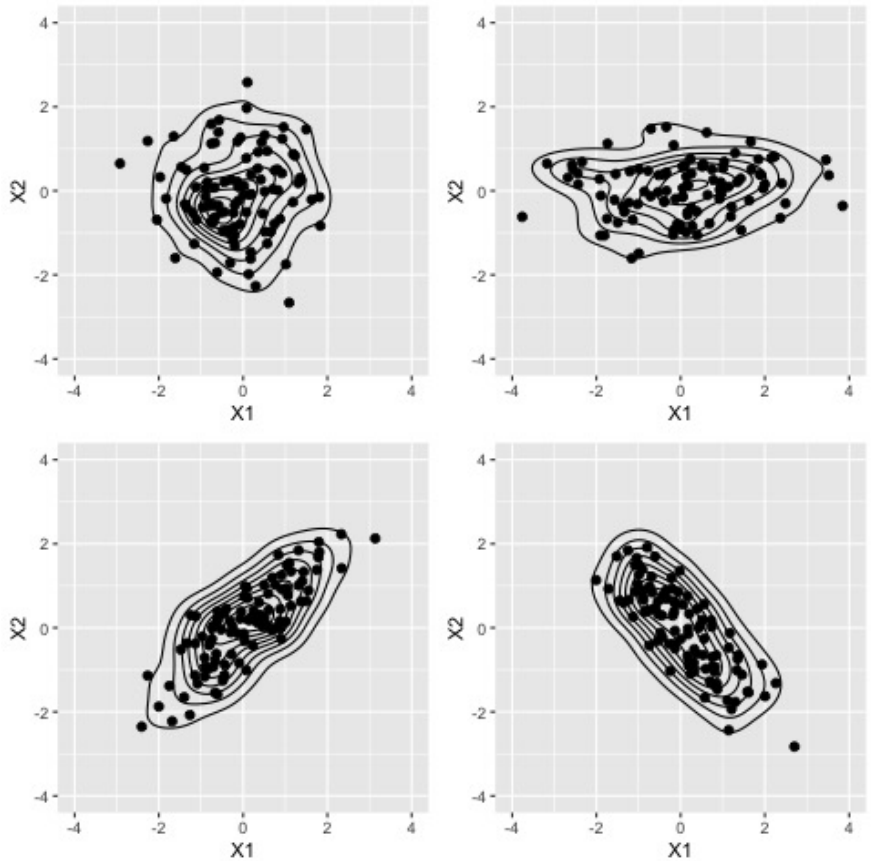
Derive the posterior probability for subject i to belong to class k based on:

- **Likelihood** of the data for p predictors (multivariate Gaussian distribution) conditional on class k

$$f(x \mid k) = \frac{1}{(2\pi)^{p/2} \mid \Sigma_k \mid^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu_k)^t \Sigma_k^{-1}(x-\mu_k)\right\}$$

  where
  - μ_k is the vector of expectations (length p) of x in group k and
  - Σ_k the covariance matrix (dimension $p \times p$) of x in group k.
  - Note that each class $k = 0$ or $k = 1$ has its specific mean μ_k and covariance Σ_k.
- **Prior probability** of belonging to class k is π_k

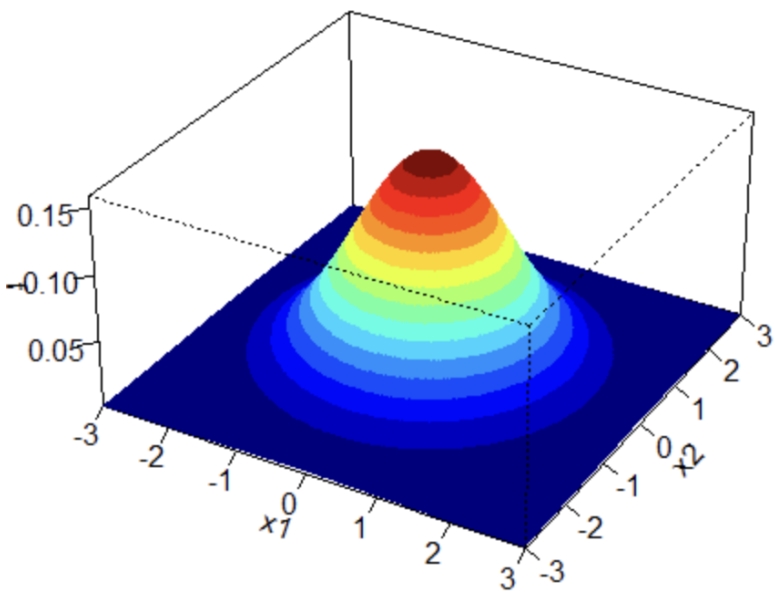
Note on the multivariate Gaussian distribution

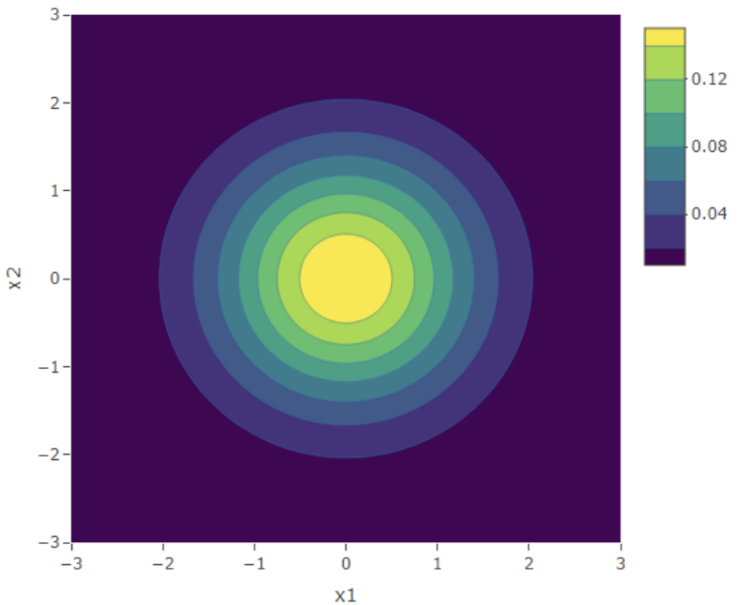


- Contour plot of two-dimensional Gaussian distribution with the same mean vector but different covariance matrices.

Note on the multivariate Gaussian distribution

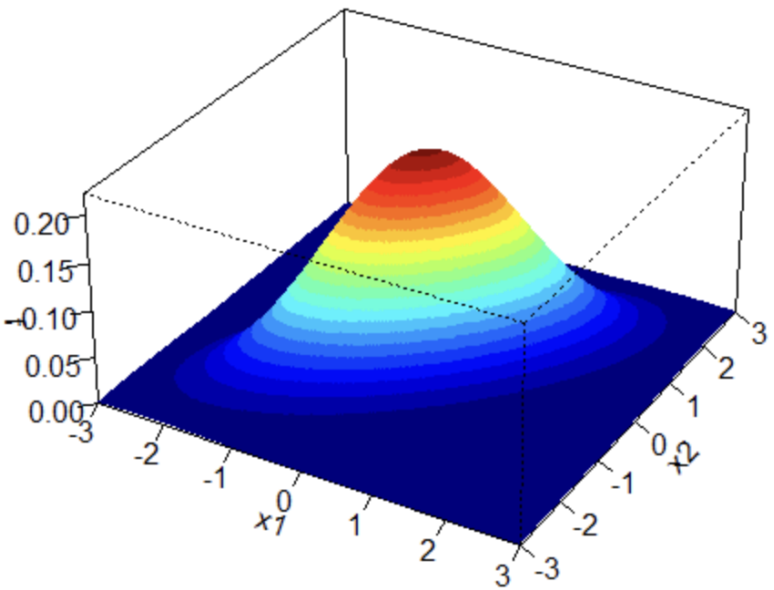
When covariance is zero

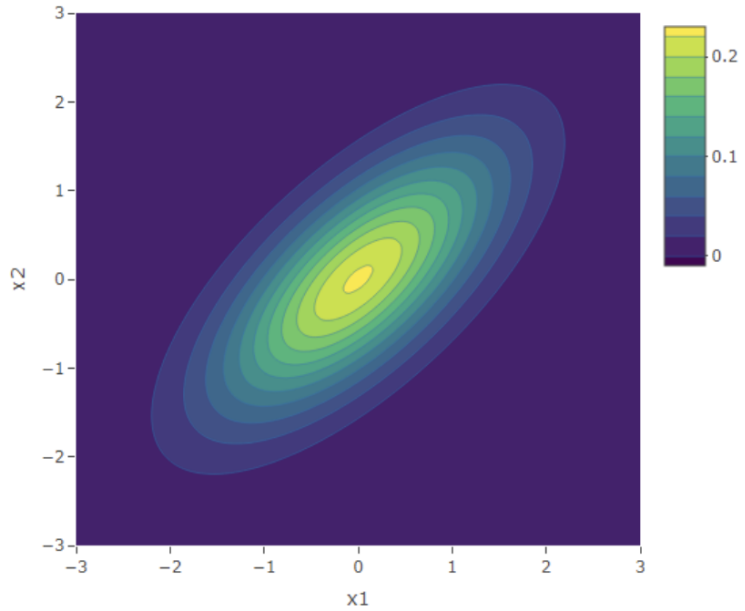




Note on the multivariate Gaussian distribution

When covariance is not zero





Bayes' Theorem for classification

Probability of being in a class or more formally: posterior probability that $y = k$, where $k = 1$ or $k = 0$, given the data x

$$\mathbb{P}(y = k \mid x) = \frac{\pi_k f(x \mid k)}{f(x)}$$

where

- π_k prior probability for class k
- $f(x \mid k)$ likelihood of x conditional on class k
- $f(x)$ joint distribution defined as mixture

$$f(x) = \pi_0 f_0(x) + \pi_1 f_1(x)$$

Prediction rule

The prediction rule is given by the difference between the two discriminant scores d_0 and d_1

$$\begin{aligned} \delta(x_i) &= \log(\mathbb{P}(y = 0 \mid x_i)) - \log(\mathbb{P}(y = 1 \mid x_i)) \\ &= d_0(x) - d_1(x) \end{aligned}$$

where observation i is classified as

- $y_i = 1$ if $\delta(x_i) \leq 0$
- $y_i = 0$ if $\delta(x_i) > 0$

Model specifications and flexibility

There are three types of discriminant analysis depending on the specification of the covariance Σ_k:

1. **Diagonal discriminant analysis (dda)**: $\Sigma_k = diag(\sigma^2)$, assuming no covariance between predictors.
   - Parameters to estimate: p diagonal elements
2. **Linear discriminant analysis (lda)**: $\Sigma_k = \Sigma$, assuming covariance between predictors, but it is the same for both groups $k = 0$ and $k = 1$.
   - Parameters to estimate: full covariance matrix ($p(p+1)/2$ parameters)
3. **Quadratic discriminant analysis (qda)**: $\Sigma_k = \Sigma_k$, assuming covariance between predictors, which may differ between group $k = 0$ and $k = 1$.
   - Parameters to estimate: twice full covariance matrix ($2 \times p(p+1)/2$ parameters)

Diagonal discriminant analysis

- **dda**: $\Sigma_k = diag(\sigma^2)$, assuming no covariance between predictors.
- dda is also known as 'naive Bayes classifier'.
- The discriminant function d_k for dda simplifies (after dropping terms which are identical for d_0 and d_1) to:

$$d_k(x) = \mu_k^t diag(\sigma^2)^{-1} x - \frac{1}{2}\mu_k^t diag(\sigma^2)^{-1}\mu_k + log(\pi_k)$$

Advantages of dda

- Easy to fit.
- Strong assumptions (no correlation between predictors and the same covariance matrix in both groups).

Linear discriminant analysis

- **lda**: $\Sigma_k = \Sigma$, allowing for covariance between predictors.
- The discriminant function d_k for lda simplifies (after dropping terms which are identical for d_0 and d_1) to:

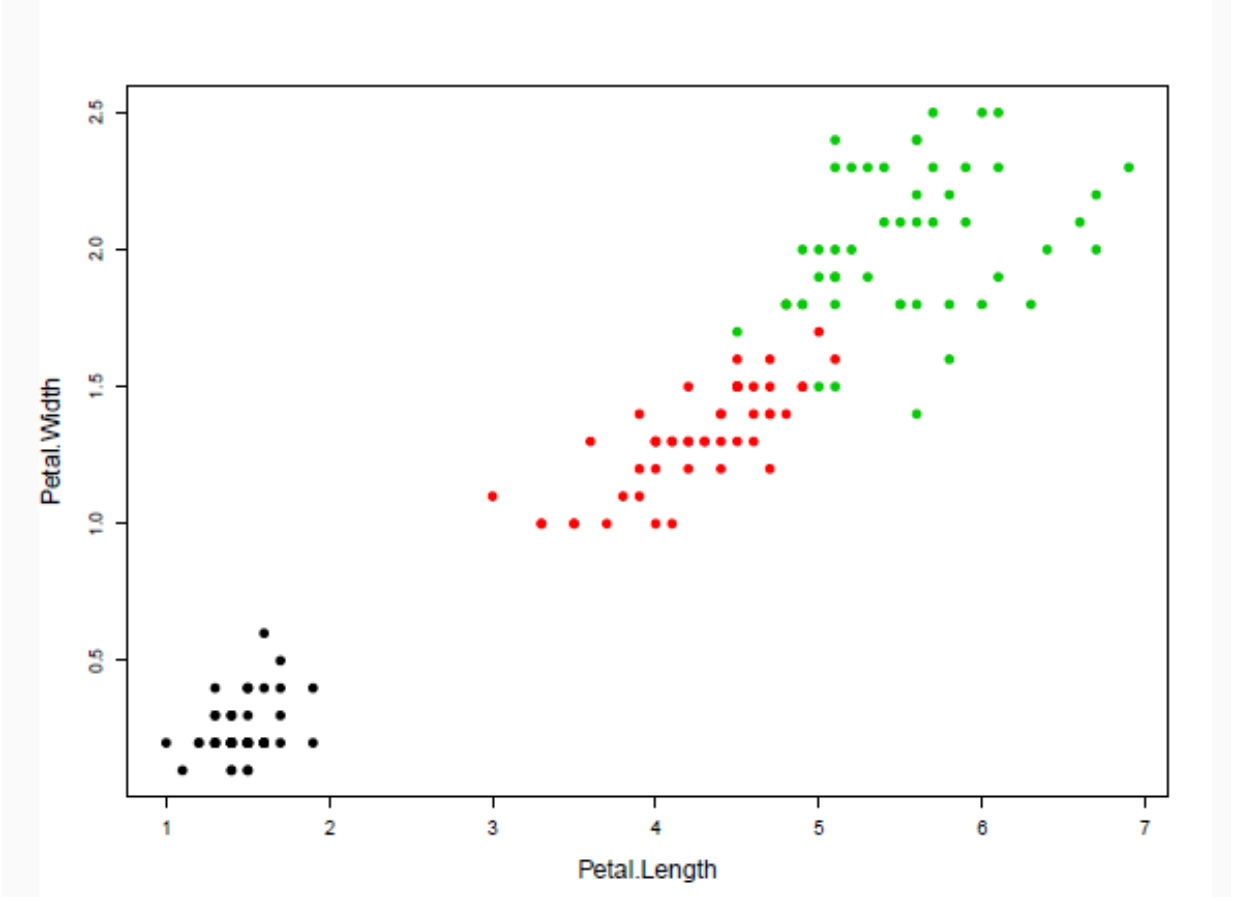
$$d_k(x) = \mu_k^t \Sigma^{-1} x - \frac{1}{2} \mu_k^t \Sigma^{-1} \mu_k + log(\pi_k)$$

Advantages of lda

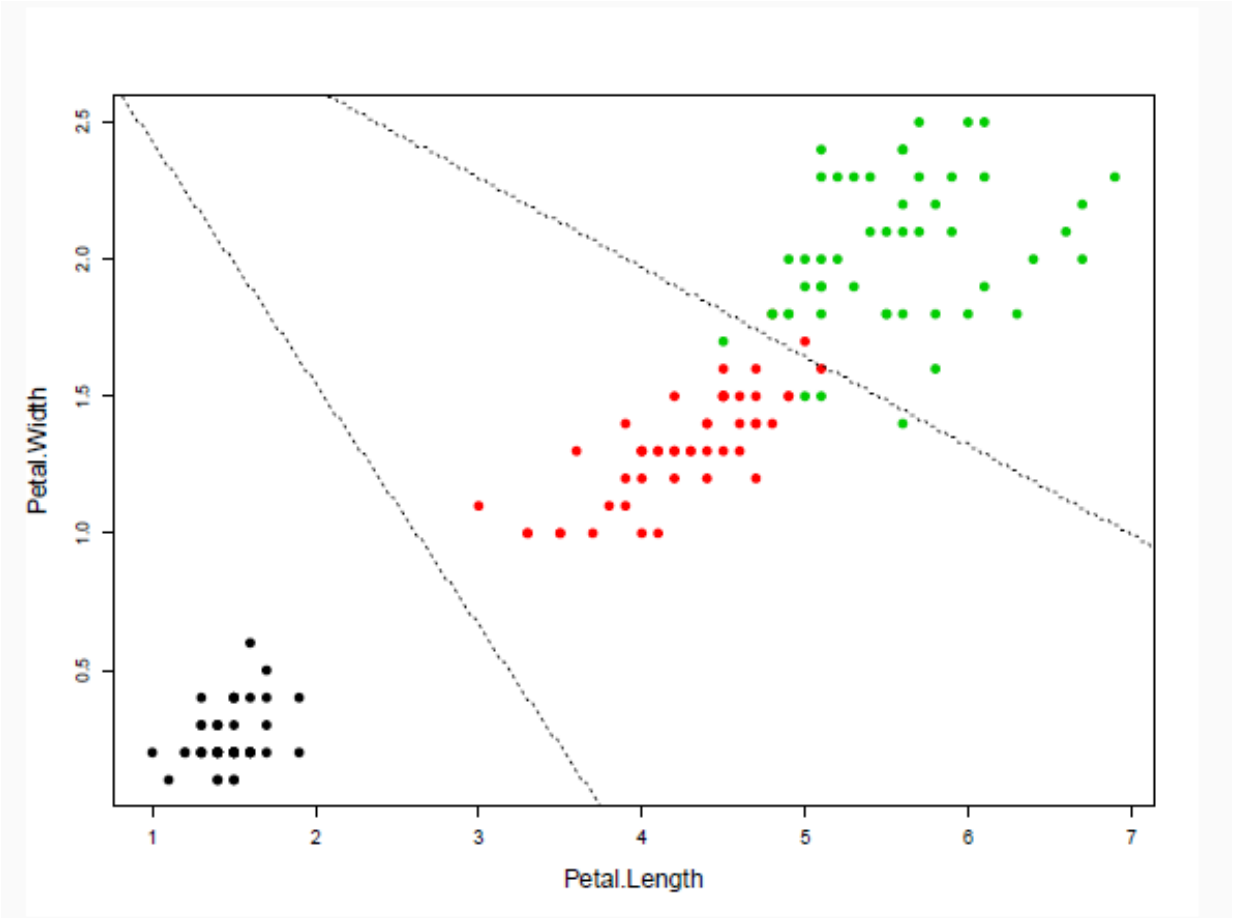
- Allowing for correlation between predictors.
- Issues for big data ($n < p$) to invert Σ.
- Assuming the same covariance matrix Σ for group $k = 0$ and $k = 1$.

Example LDA

- Iris dataset: separate the different species based on petal width and length



Example LDA



Quadratic discriminant analysis

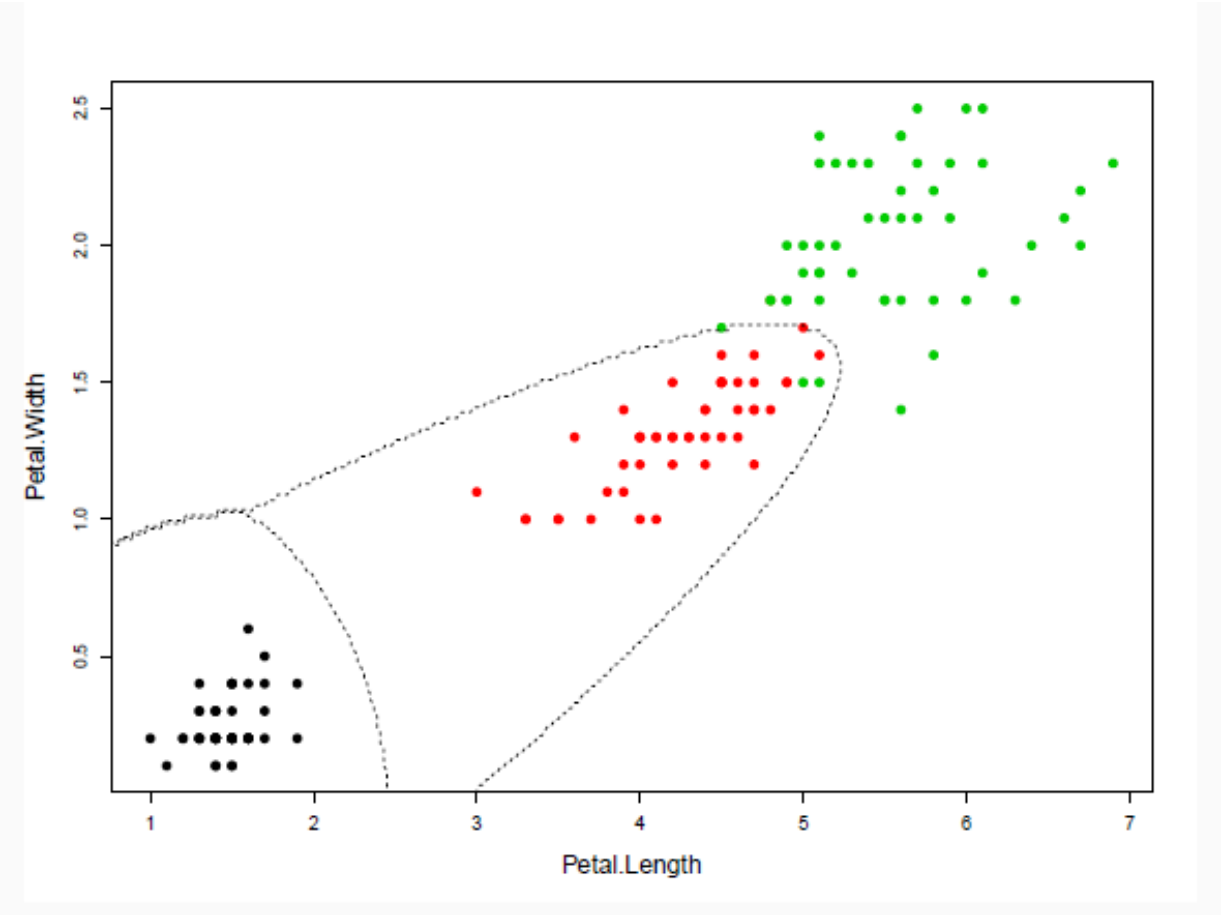
- **qda**: Σ_k, allowing for different covariances between groups.
- The discriminant function d_k for qda simplifies (after dropping terms which are identical for d_0 and d_1) to:

$$d_k(x) = \mu_k^t \Sigma_k^{-1} x - \frac{1}{2} \mu_k^t \Sigma_k^{-1} \mu_k + log(\pi_k)$$

Advantages of lda

- Allowing for correlation between predictors.
- Issues for big data ($n < p$) to invert Σ.
- qda allows for the most flexibility at the cost of many additional parameters.

QDA Iris



lda in the MASS package: Iris

```
head(iris)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa

```
#fit LDA model
model <- lda(Species~., data=train)
model
```

lda in the MASS package: Iris

```
Call:
lda(Species ~ ., data = train)

Prior probabilities of groups:
    setosa versicolor  virginica
 0.3207547  0.3207547  0.3584906

Group means:
           Sepal.Length Sepal.Width Petal.Length Petal.Width
setosa       -1.0397484   0.8131654   -1.2891006  -1.2570316
versicolor    0.1820921  -0.6038909    0.3403524   0.2208153
virginica     0.9582674  -0.1919146    1.0389776   1.1229172

Coefficients of linear discriminants:
                   LD1         LD2
Sepal.Length  0.7922820   0.5294210
Sepal.Width   0.5710586   0.7130743
Petal.Length -4.0762061  -2.7305131
Petal.Width  -2.0602181   2.6326229
```

lda in the MASS package

- Fit lda classification rule: `lda.out = lda(x, y)`

```
Call:
lda(x, grouping = y)

Prior probabilities of groups:
  0   1
0.5 0.5

Group means:
           1          2
0 0.02802684 0.09566739
1 1.97080059 1.89748769

Coefficients of linear discriminants:
           LD1
[1,] 0.7079785
[2,] 0.7620127
```

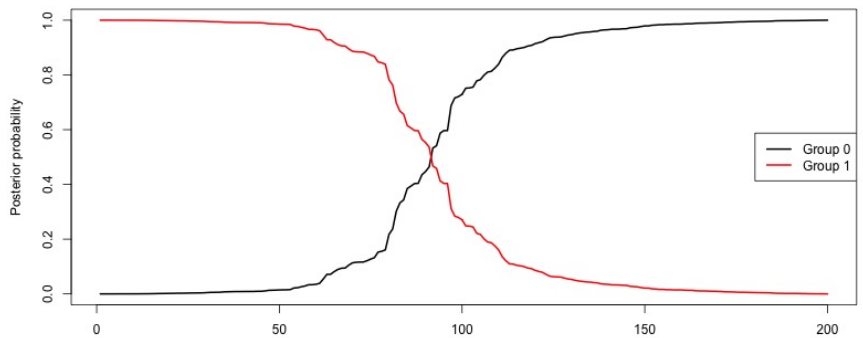
- Equivalently, use `qda.out = qda(x, y)` for qda.
- Prediction: `lda.pred = predict(lda.out, x)`
  - lda.pred$class: predicted class

```
> table(lda.pred$class)

  0   1
109  91
```

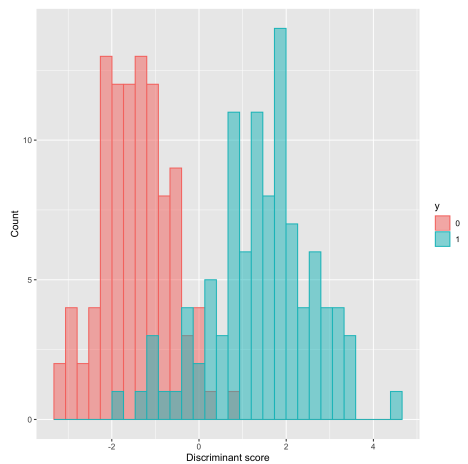
lda in the MASS package

Prediction: `lda.pred = predict(lda.out, x)`

- lda.pred$posterior: posterior probabilities for group 0 and 1



- lda.pred$x: discriminant score



Shrinkage estimate

- Standard implementations of discriminant analysis use the sample covariance to estimate Σ.
- The computation of the discriminant function for lda and qda requires the inversion of Σ, a $p \times p$ matrix.
- In high-dimensional settings where $n < p$, Σ cannot be inverted.
- Shrinkage estimates for Σ ensure lda and qda can be computed for high-dimensional data.
- Implementation `sda` package in `R`.

Let me know if you want more references and slides here.

Application: Imaging data to predict breast cancer diagnosis

- Outcome: Diagnosis (M = malignant, B = benign)

```
> table(y)
y
  B   M
357 212
```

- $n = 569$ Breast cancer patients
- $p = 30$ Predictors:
  - Derived from digitized images to define characteristics of the cell nuclei present in the image.
  - Ten real-valued features summarised in mean, se and worst (mean of the three largest values).
- Dataset from the UCI Machine Learning Repository: `http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29`

sda package in R

- dda: sda(x, y, diagonal=TRUE)

```
> dda.out = sda(x, y, diagonal=TRUE)
Number of variables: 30
Number of observations: 569
Number of classes: 2

Estimating optimal shrinkage intensity lambda.freq (frequencies): 0.0254
Estimating variances (pooled across classes)
Estimating optimal shrinkage intensity lambda.var (variance vector): 0.0186

> dda.pred = predict(dda.out, x, verbose=FALSE)
```

- confusionMatrix from the crossval package

```
> cM=confusionMatrix(y, dda.pred$class, negative="B")
> cM
 FP  TP  TN  FN
  6 185 351  27
attr(,"negative")
[1] "B"
> TPR = cM[2]/(cM[2]+cM[4])
> TPR
       TP
0.8726415
> TNR = cM[3]/(cM[1]+cM[3])
> TNR
       TN
0.9831933
```

sda package in R

- lda: sda(x, y, diagonal=FALSE)

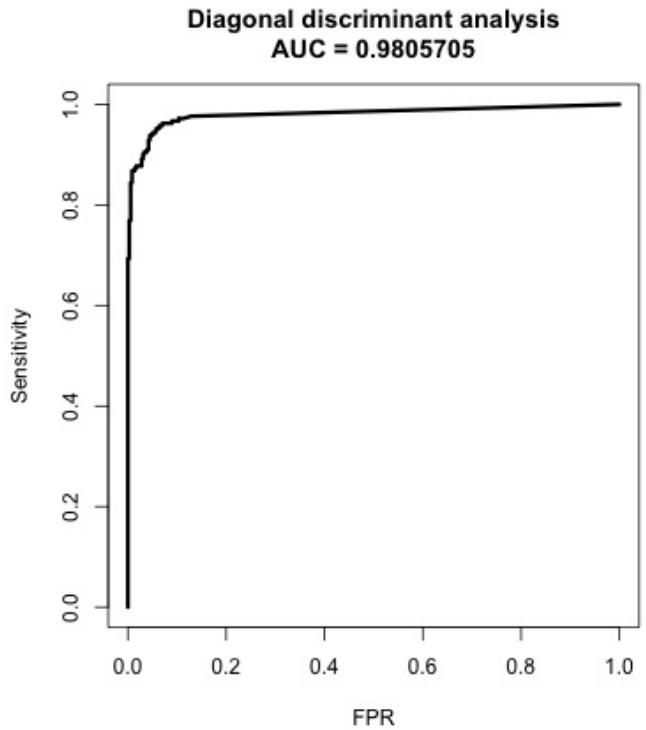
```
> lda.out = sda(x, y, diagonal=FALSE)
Number of variables: 30
Number of observations: 569
Number of classes: 2

Estimating optimal shrinkage intensity lambda.freq (frequencies): 0.0254
Estimating variances (pooled across classes)
Estimating optimal shrinkage intensity lambda.var (variance vector): 0.0186


Computing inverse correlation matrix (pooled across classes)
Estimating optimal shrinkage intensity lambda (correlation matrix): 0.0314
> lda.pred = predict(lda.out, x, verbose=FALSE)
```

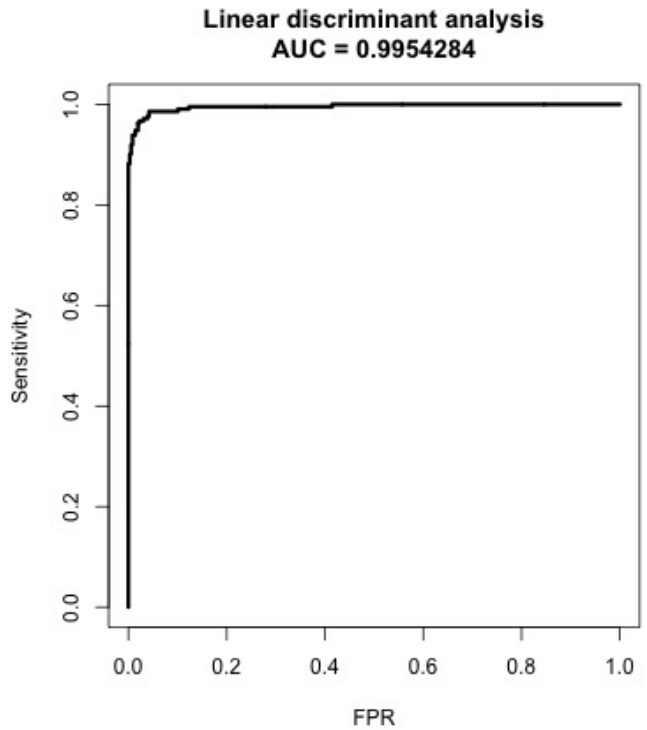
- confusionMatrix from the crossval package

```
> cM=confusionMatrix(y, lda.pred$class, negative="B")
> cM
 FP  TP  TN  FN
  2 193 355  19
attr(,"negative")
[1] "B"
> TPR = cM[2]/(cM[2]+cM[4])
> TPR
       TP
0.9103774
> TNR = cM[3]/(cM[1]+cM[3])
> TNR
       TN
0.9943978
```

ROC curve

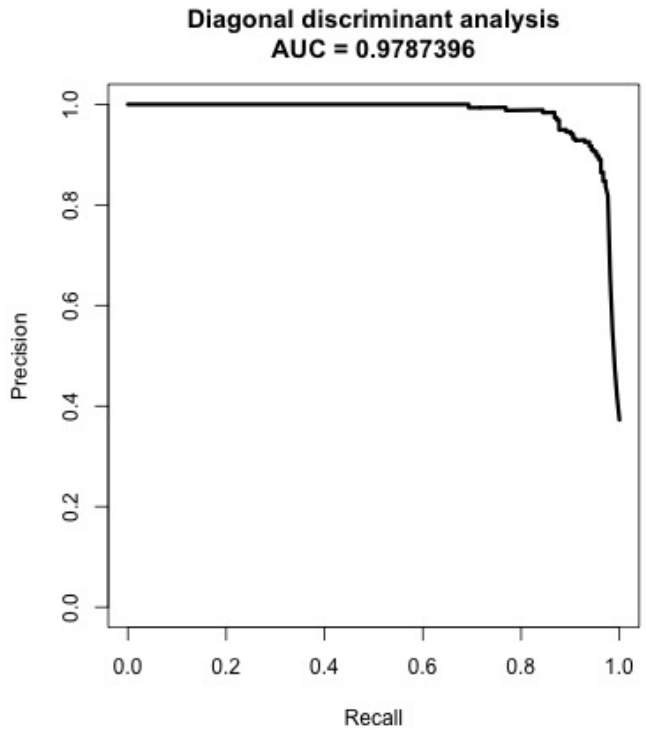
1. Define foreground (fg) and background (bg):
   ```
   fg = dda.pred$posterior[y == 'M',2]
   bg = dda.pred$posterior[y == 'B',2]
   ```
2. Compute the ROC curve:
   ```
   roc.dda = roc.curve(fg,bg,curve=T)
   ```

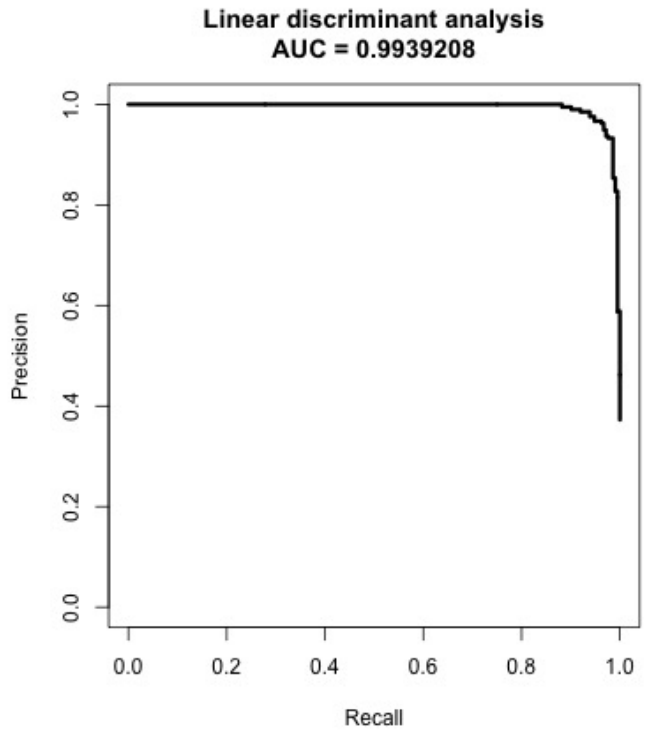




Precision-recall curve

1. Define foreground (fg) and background (bg):
   ```
   fg = dda.pred$posterior[y == 'M',2]
   bg = dda.pred$posterior[y == 'B',2]
   ```
2. Compute the ROC curve:
   ```
   pr.dda=pr.curve(fg,bg,curve=T)
   ```





An introduction to support vector machines

Evolution of support vector machines:

1. **Hyperplanes** to introduce simple decision boundaries.
2. **Maximal margin classifier** as the 'best' hyperplane.
3. **Support vector classifier** allowing for soft margins.
4. **Support vector machine** using kernels for non-linear decision boundaries.

What is a hyperplane?

- In a p-dimensional space, a hyperplane is a flat affine subspace of dimension $p - 1$.
  $\rightarrow$ Considering two dimensions, a hyperplane is a line.
  $\rightarrow$ Considering three dimensions, a hyperplane is a plane.
- For example in two dimensions, a hyperplane is defined by

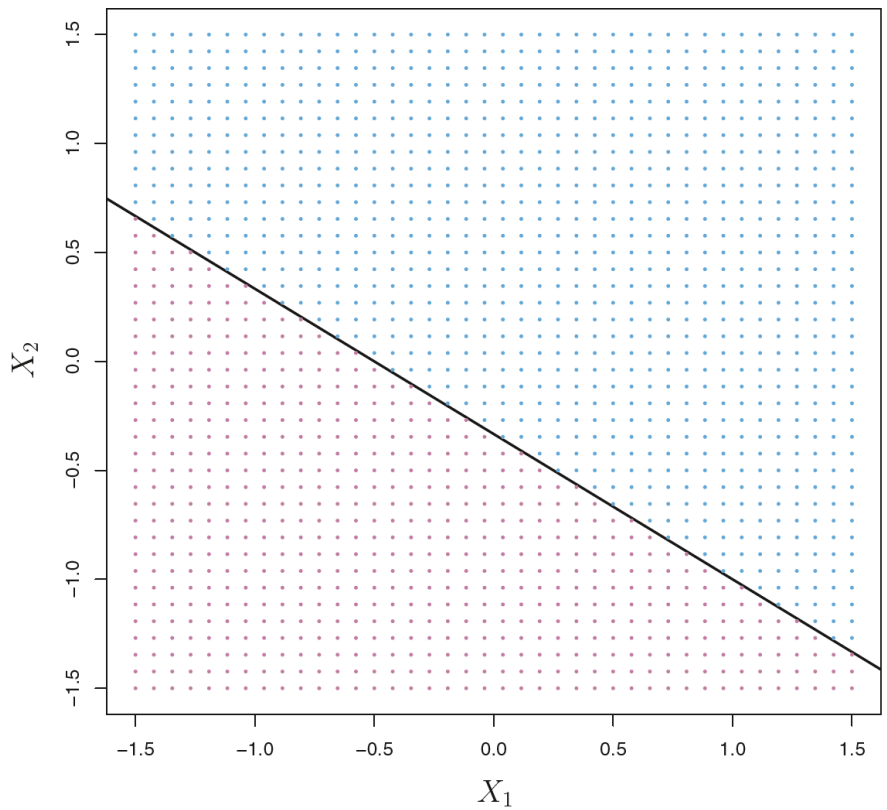
$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0$$

- A hyperplane separates observations $x = (x_1, x_2)$ into two groups

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 \begin{cases} > 0 \\ < 0 \end{cases}$$

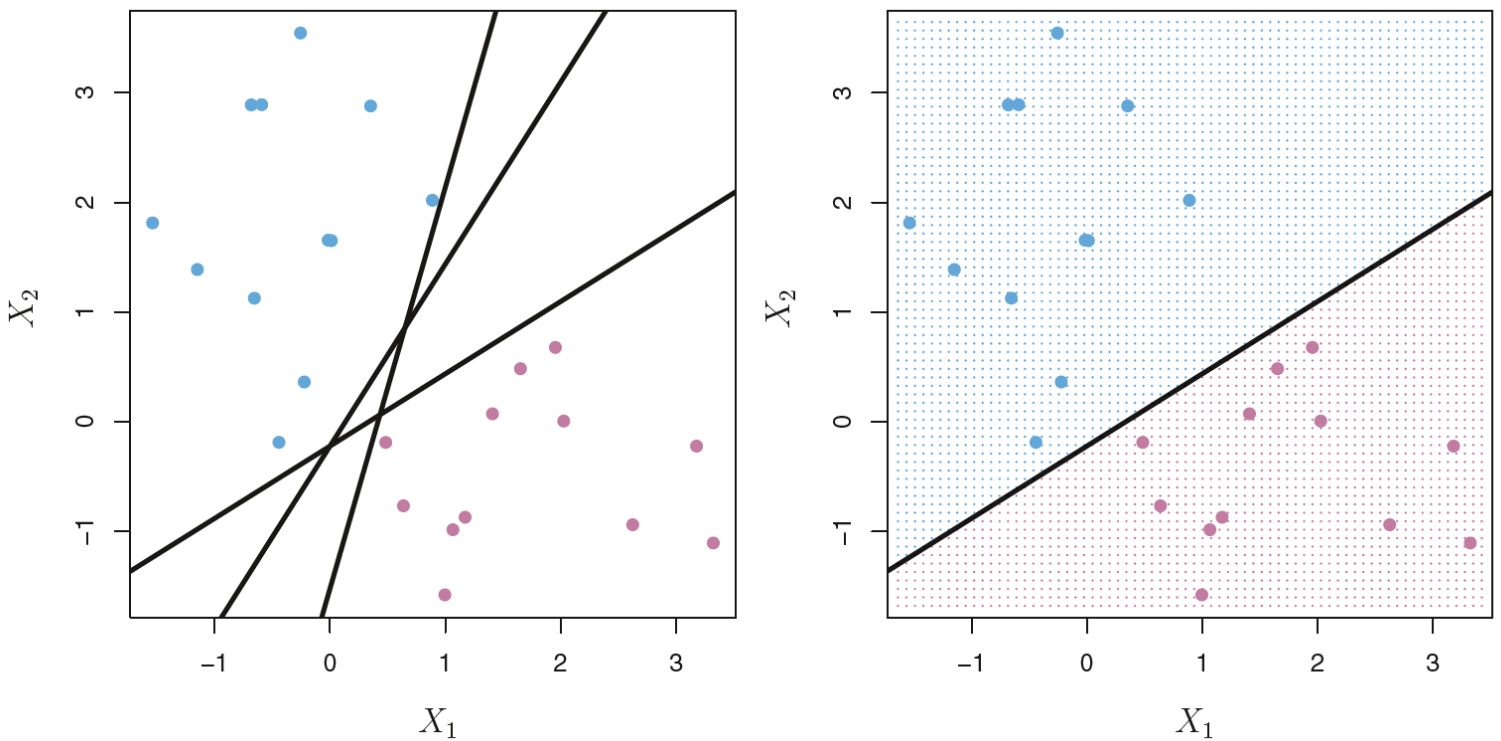
What is a hyperplane?

$$1 + 2x_1 + 3x_2 \begin{cases} > 0 \text{ blue} \\ < 0 \text{ purple} \end{cases}$$



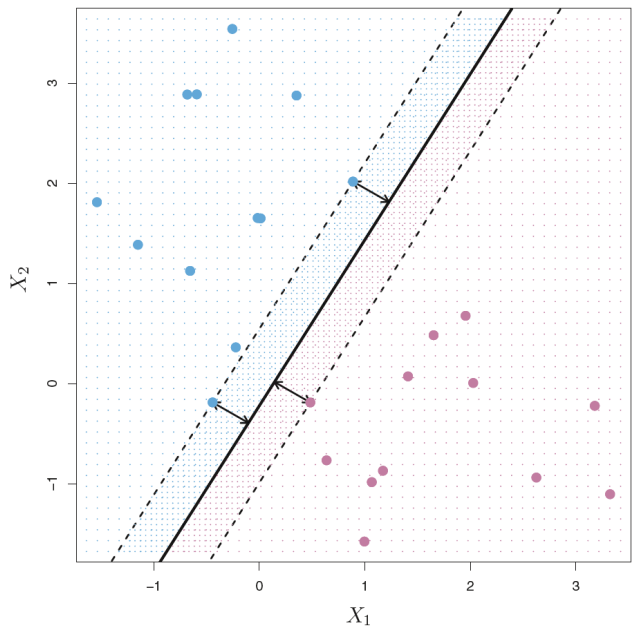
What is a hyperplane?

- If two classes are perfectly separable, there is not a unique solution for a hyperplane.



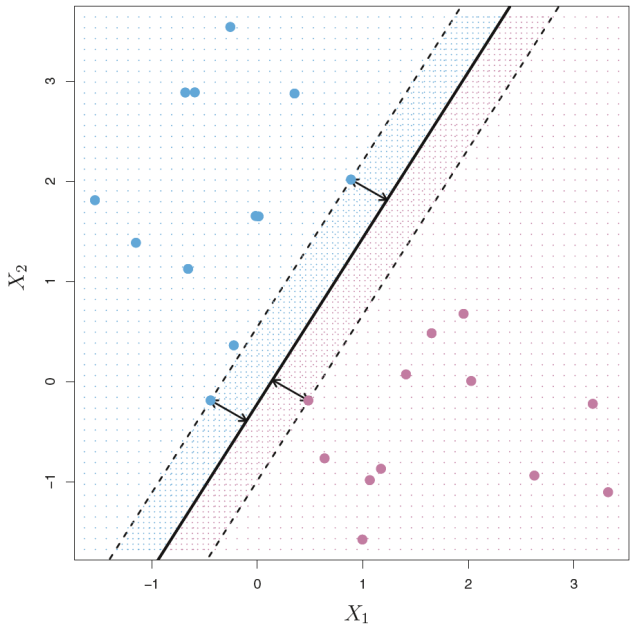
Maximal margin classifier

- Maximal margin classifier provide unique analytical solutions.
- This is the optimal separating hyperplane that has the highest minimum distance (margin) to the training data.



- Note: The maximal margin classifier only depends on the three observations closest to the decision boundary, not to the other observations.

Maximal margin classifier



We need to define

1. Separating hyperplane $\beta_0 + \beta_1 X1 + \beta_2 X_2$
2. Margin (distance to the closest data points from hyperplane)
3. Support vectors (points with distance equal to the margin from hyperplane)

Optimisation: Maximal margin classifier

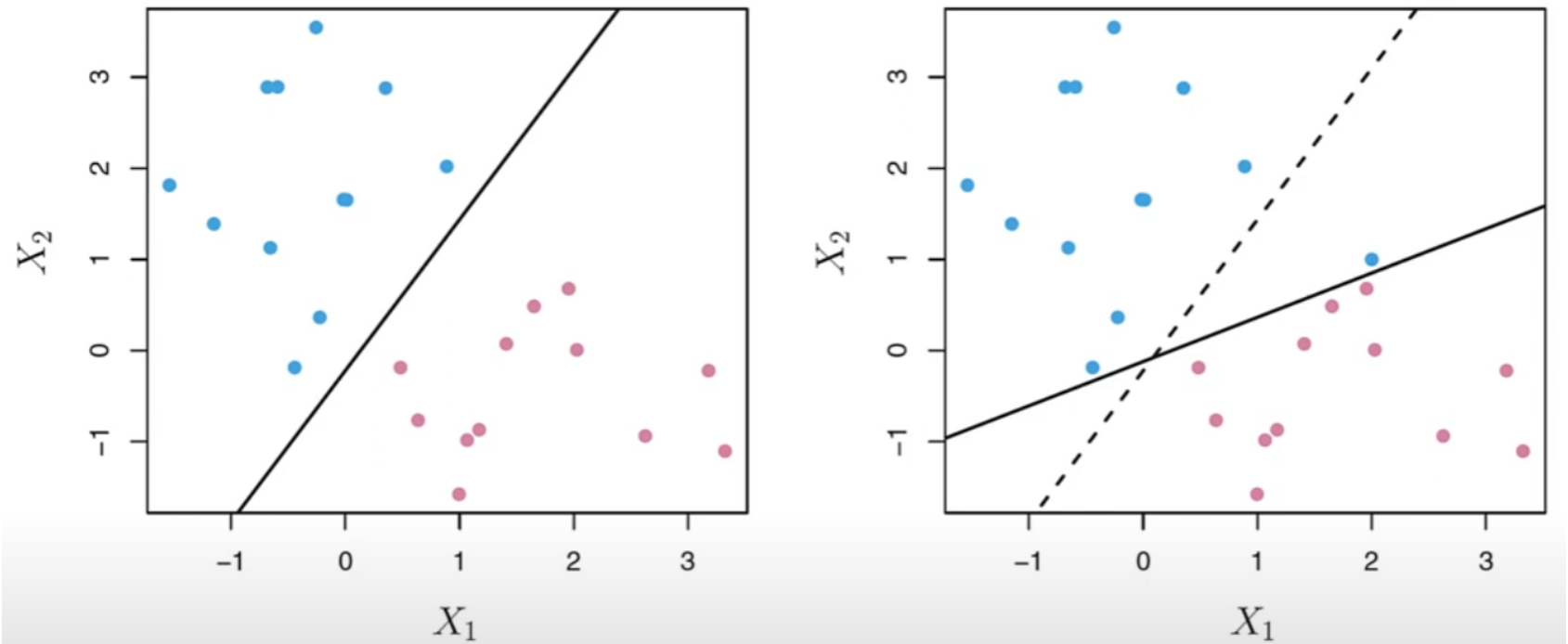
$$\max_{\beta_0,\beta_1,...,\beta_p} M$$

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1 \text{ forces unique sollution}$$

$$y_i(\beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip}) \geq M \quad \forall i \in 1, ..., n$$

- $\beta_0, \beta_1, ..., \beta_p$ parameters to be fitted
- $i \in 1, ..., n$ observations
- M is the margin, distance of observation to decision boundary
- Outcome: $y_i = \begin{cases} 1 & \text{if } i \text{ in group 1} \\ -1 & \text{if } i \text{ in group 0} \end{cases}$

Maximal margin classifier



- Good for separating non-Gaussian data
- Is not possible if there is no linear separation

Support vector classifier

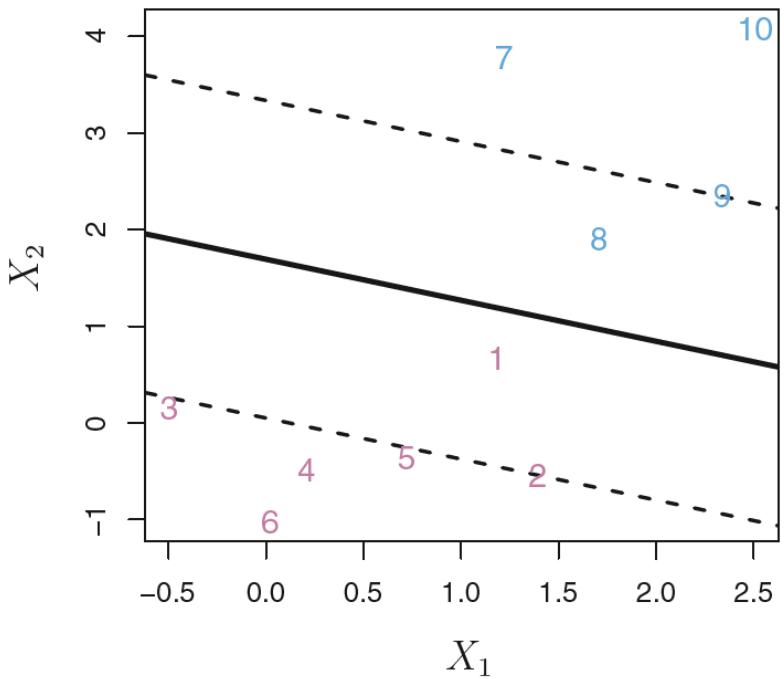
- **Support vectors**: Observations which support the maximal margin hyperplane.
- In most application examples, there is no separating hyperplane that can separate two groups.

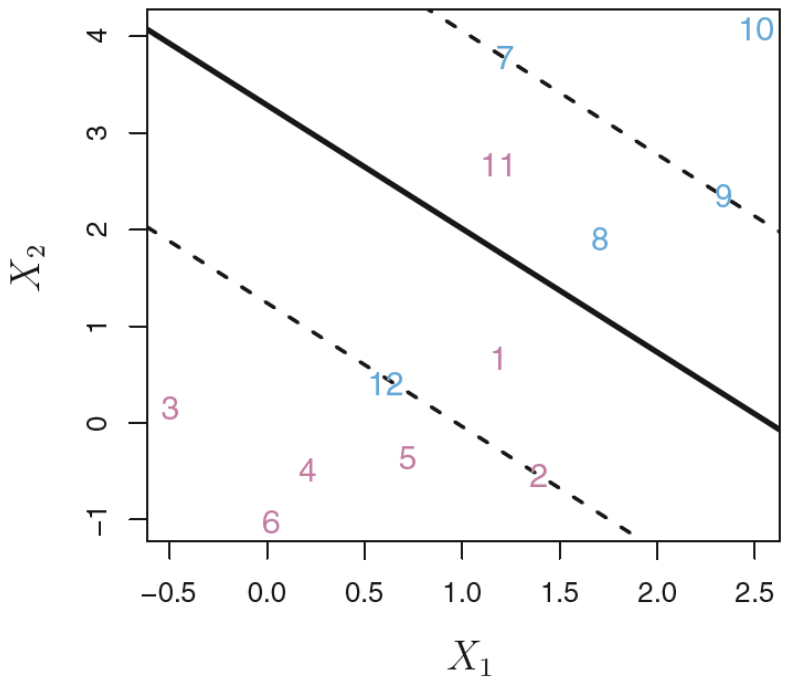
Aim: Support vector or soft margin classifier

- Develop a hyperplane that almost separates the classes.
- Better classification of majority of observations.
- Greater robustness to individual variation.
- Avoid overfitting to the training data.

Distinction: Margin and hyperplane

- **Hyperplane**: Decision boundary (solid line).
- **Margin**: Optimised in maximal margin classifier (dashed line).





Optimisation: Support vector classifier

$$\max_{\beta_0,\beta_1,\ldots,\beta_p} M \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i) \quad \forall i \in 1, \ldots, n$$

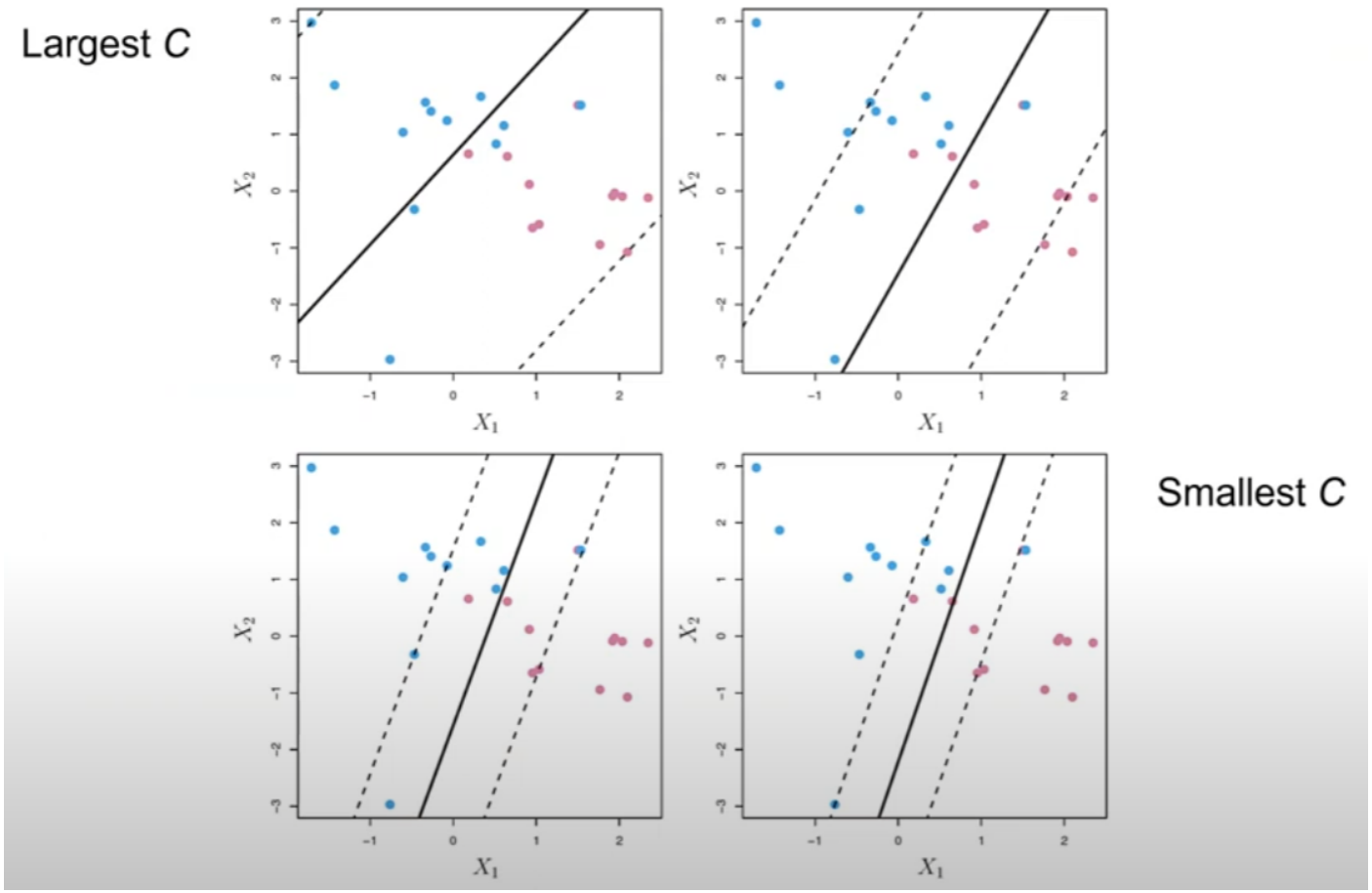
$$\text{where } \epsilon_i \geq 0 \text{ and } \sum_{i=1}^{n} \epsilon_i \leq C$$

- **Slack variables** allow for a few observations to be on the wrong side of the margin or the hyperplane (missclassification)
- C: Budget for the amount that the margin can be violated.

Difference to maximal margin classifier:

- Regularisation in form of a tuning parameter C and slack variables introduces bias but provides better generalisation.

Support vector classifier



Support vector machines

- Support vector classifier assume linear decision boundaries.
- How can we allow for more flexibility and non-linear decision boundaries?

1st Idea: Include quadratic, cubic, or higher-order polynomial functions of the predictors.

- Add the squared $x_1^2, ..., x_p^2$ or cubic predictors $x_1^3, ..., x_p^3$.

2nd Idea: Use kernel functions.

- The optimization algorithm of the support vector classifier is based on the dot product between two observations.

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

Kernel functions

The dot product is also known as linear kernel

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}.$$

Instead of the linear kernel other kernel functions may be used:

- Polynomial kernel:

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d$$

- Radial basis:

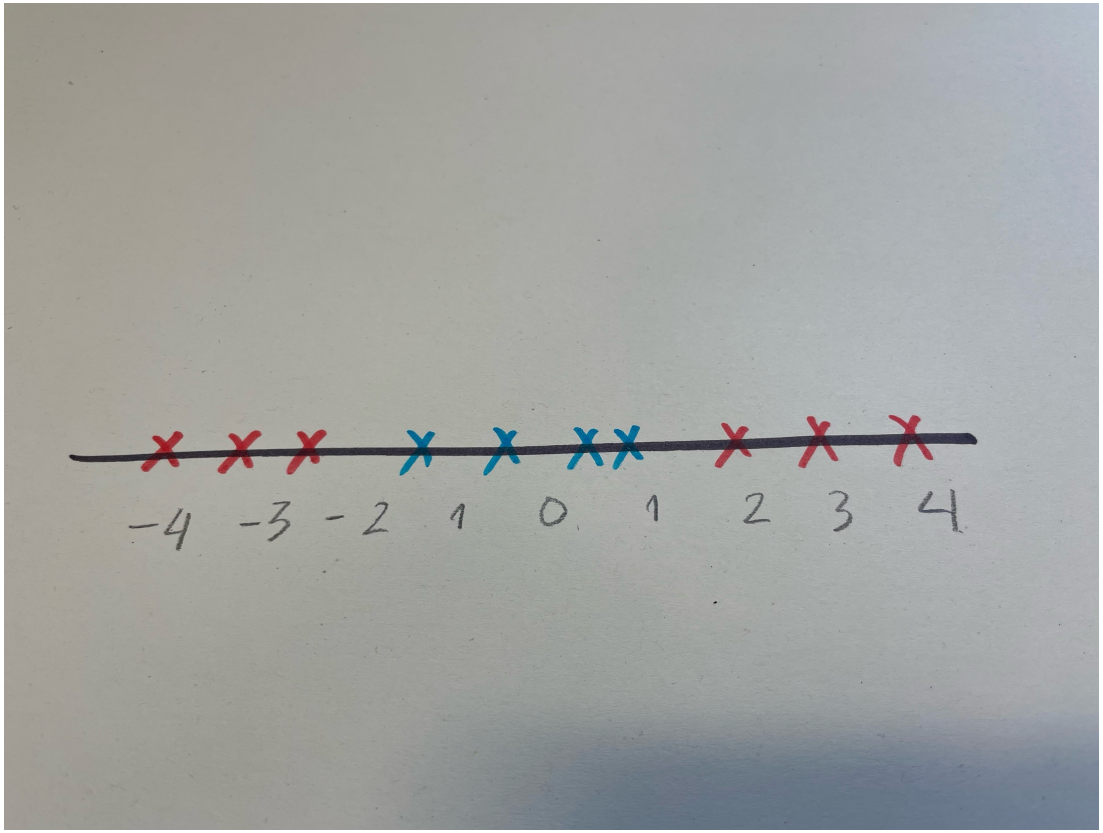
$$K(x_i, x_{i'}) = exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2)$$

- Sigmoid kernel:

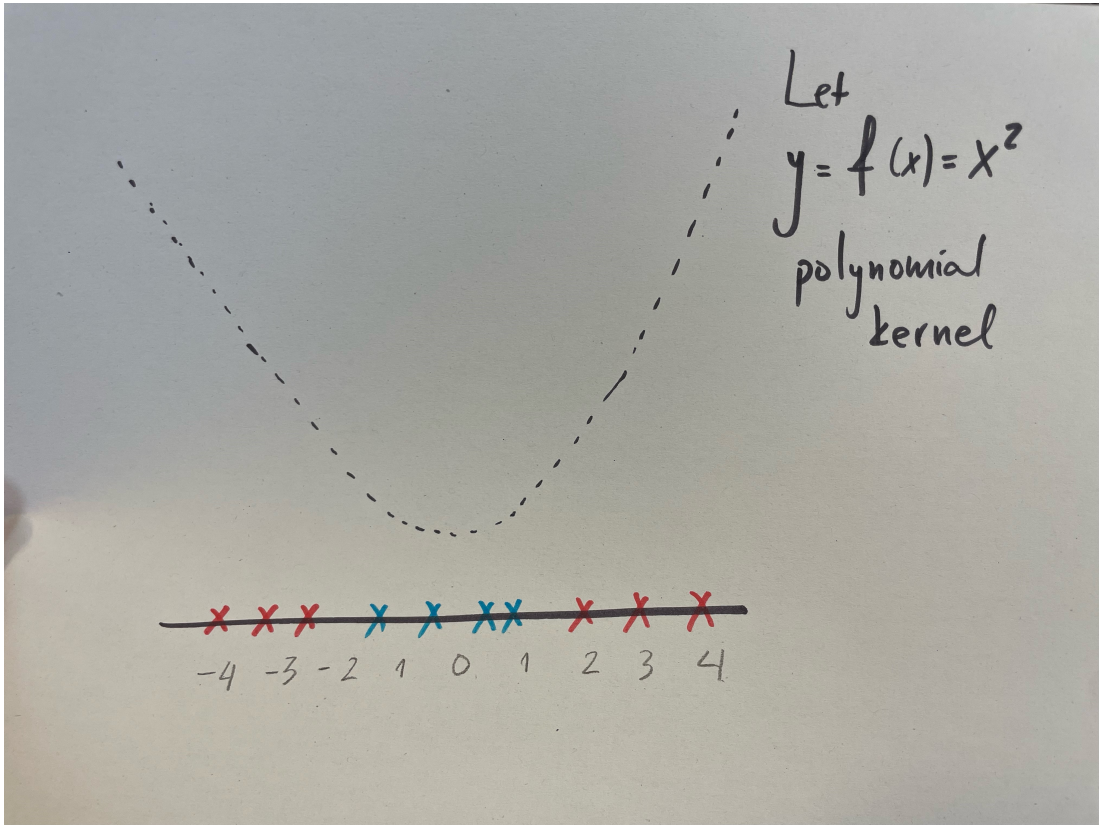
$$K(x_i, x_{i'}) = \gamma \sum_{j=1}^{p} x_{ij} x_{i'j} + r$$

Polynomial kernel of degree 2 example

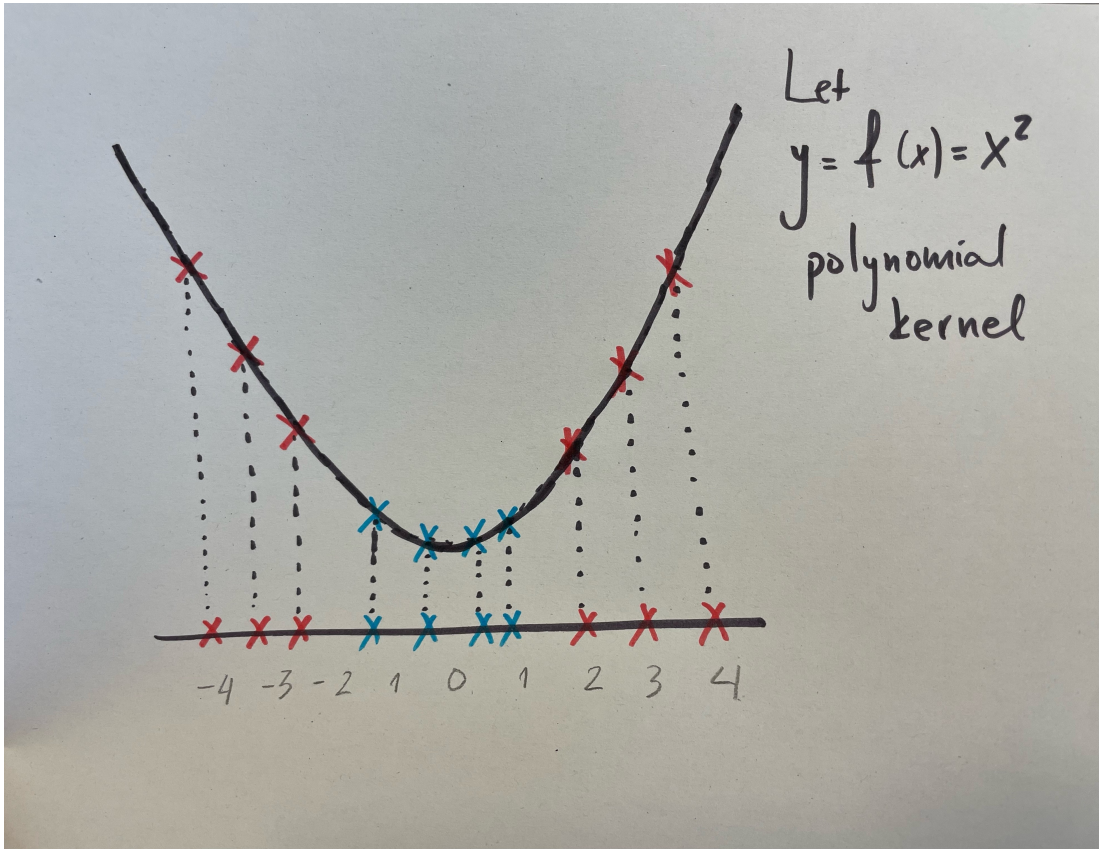
Idea: Convert to higher dimension and separate using a hyperplane



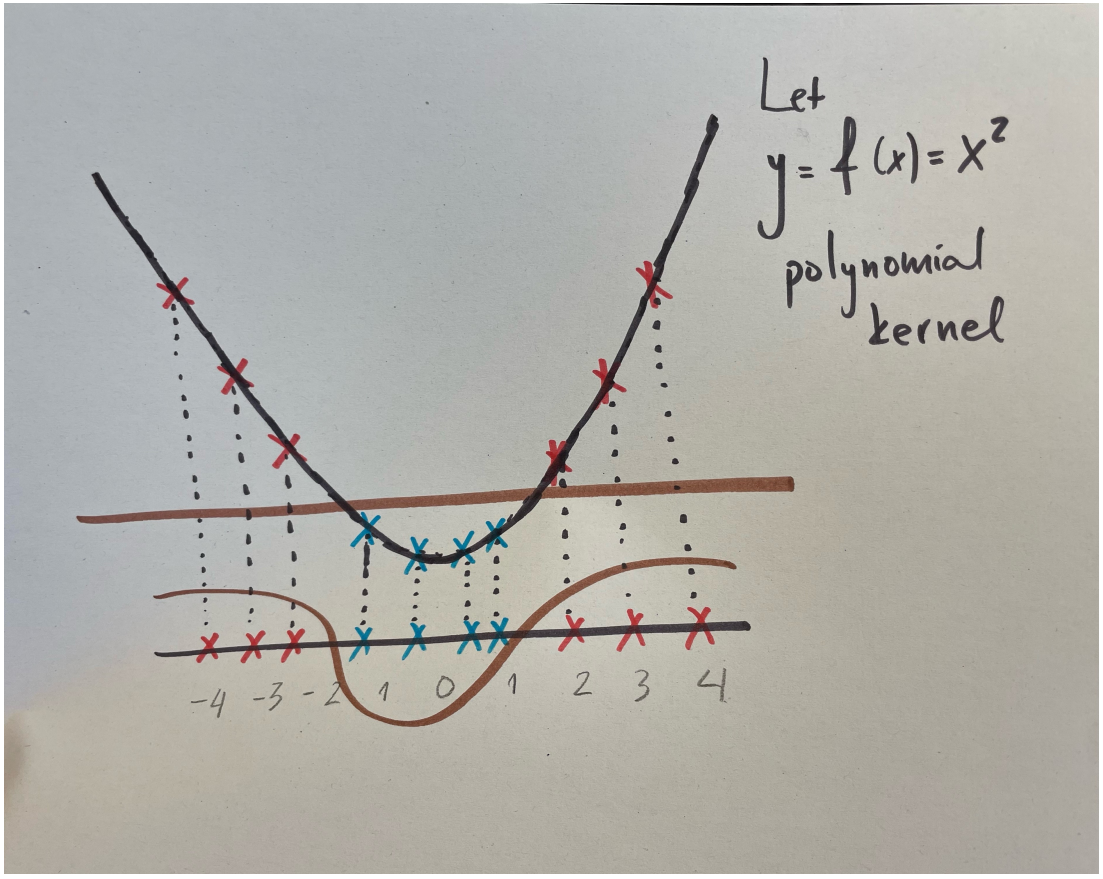
Polynomial kernel of degree 2 example



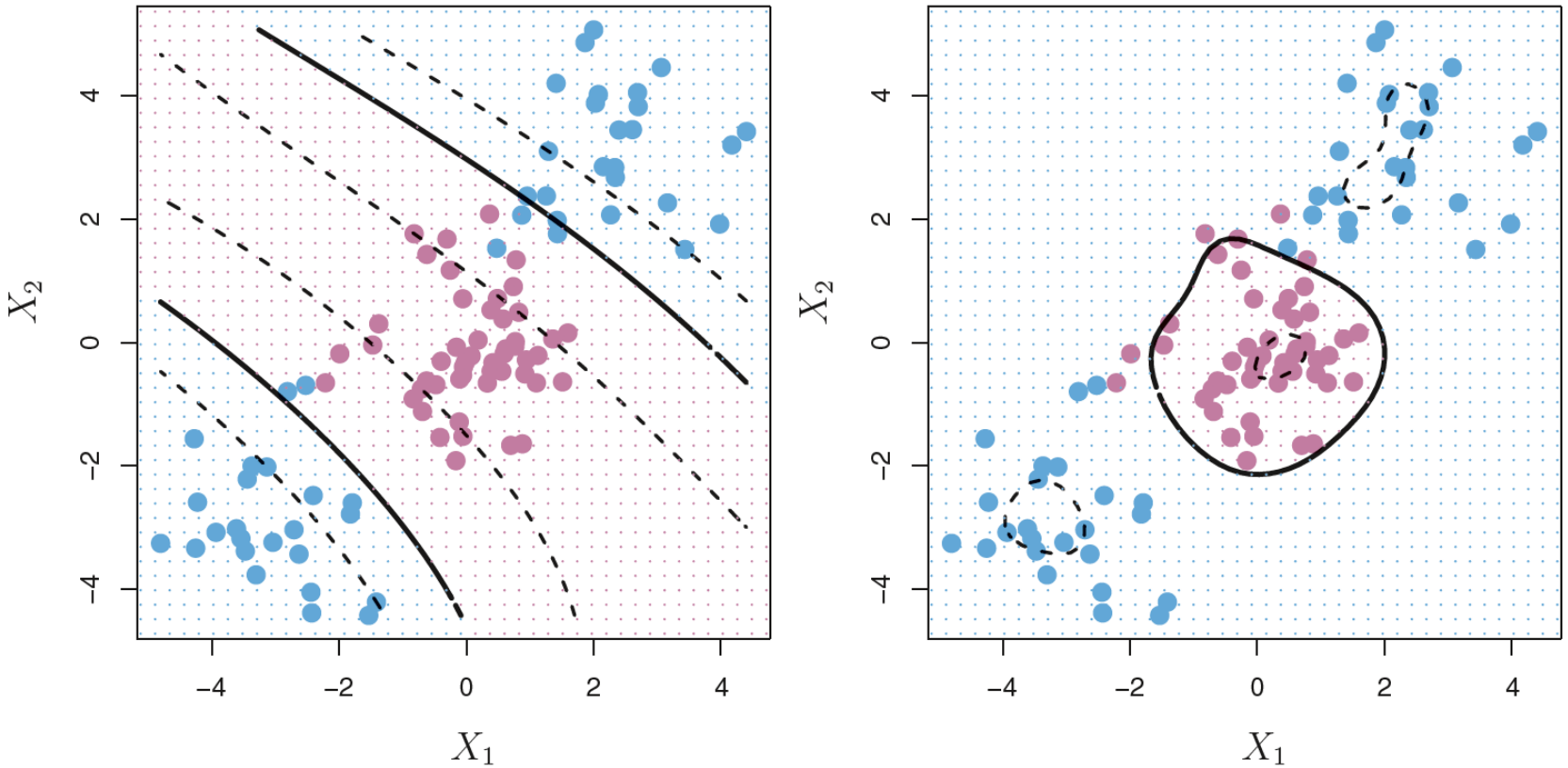
Polynomial kernel of degree 2 example



Polynomial kernel of degree 2 example



Kernel functions



- Left: Polynomial kernel of degree 3
- Right: Radial kernel

svm function in R

- svm function in the library(e1071)

```
> svm.out = svm(x, y)
> svm.out

Call:
svm.default(x = x, y = y)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  119

>
> svm.pred = predict(svm.out, x)
> table(svm.pred)
svm.pred
  B   M
364 205
```

svm function in R

```
> cM=confusionMatrix(y, svm.pred, negative="B")
> cM
 FP  TP  TN  FN
  0 205 357   7
attr(,"negative")
[1] "B"
> TPR = cM[2]/(cM[2]+cM[4])
> TPR
       TP
0.9669811
> TNR = cM[3]/(cM[1]+cM[3])
> TNR
TN
 1
```

Options:

- `kernel`: Kernel function
- `cost`: Regularisation parameter C or budget
- `probability = TRUE`: To output posterior probabilities of r class membership

Take away: Machine learning: Classification

- Evaluation of classification performance based on parameters derived from the confusion matrix.
- Visualisation using ROC and precision-recall curves.
- Discriminant analysis is a parametric approach for classification.
- Diagonal, linear and quadratic discriminant analysis are based on different assumptions on the covariance matrix and allow for different flexibility in model fit.
- Support vector machines are non-parametric machine-learning type of approaches which offer the most flexibility.
- Support vector machines are optimized for prediction and operate in a black box type of implementation (hard to understand mechanistics and interpret).

General background reading

An Introduction to Statistical Learning

with Applications in R

Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani

Home

About this Book

R Code for Labs

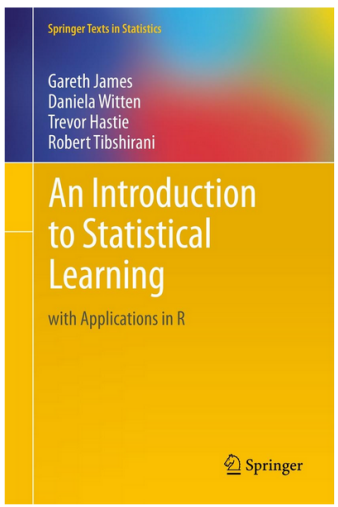
Data Sets and Figures

ISLR Package

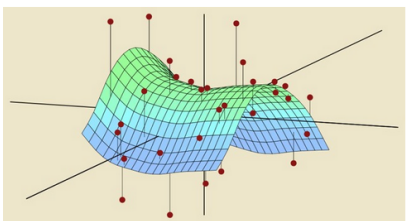
Get the Book

Author Bios

Errata



Download the book PDF
(corrected 7th printing)



Statistical Learning MOOC covering the entire ISL book offered by Trevor Hastie and Rob Tibshirani. Start anytime in self-paced mode.

An Introduction to Statistical Learning:

- 4.4 Linear Discriminant Analysis
- 9 Support Vector Machines

`http://faculty.marshall.usc.edu/gareth-james/ISL/`

Next lectures

LECTURE 4b Machine learning: Ensemble methods

- Decision trees
- Ensemble methods

LECTURE 4c Machine learning: Neural networks

- Neural networks
- Deep learning
- Machine learning resources in R